Lab5 PHP_PDO Lab

Table of Contents

Goal 2

Lab tasks 2

Lab Report: 2

Recommended Software 2

How to screenshot the active application 2

Resources about how to work with database in PHP: 2

1. *Let's get tables and data ready.* 4
2. *Get the login function to work with the real user and passwd in the database* 4
3. *Implement the function of displaying balance in bankoperation.php* 7
4. *Implement the transfer function* 9

Goal

In this lab, you will practice the following.

Database programming in PHP

a. Connect to database
b. Retrieve (select) and update data (update and delete)
c. Use prepared statement
d. Use transaction

Lab tasks

In this lab, we will finish the last two parts for our mini back that we started in Lab4. We separate the implementation of the mini bank system into different modules:

1. Authentication module (login.php)
2. Main control module (main.php)
3. Business module (bankoperation.php)
4. Database Operation module (db.php)

Lab Report:

You don't need to write report for each step. At the end of the lab session, in your report include the url/screenshot/source as listed in the report template file.

1. login.php
2. bankoperation.php
3. db.php source code

Recommended Software

Same as Lab4.

- BIG_IP_Edge Client (VPN client)
- Multi Drive (Mount the M:drive /local/my_web_files/your_user_name) on to your local computer)
- Visual Studio Code (Edit html and php files)

How to screenshot the active application

Windows: Alt+PrtSc The screenshot will be save to the clipboard.
or Win + Shift + S keys for specific areas.

Mac: Command+Shift+4, then spacebar. then your mouse cursor will turn into a camera icon. Move the camera icon to the active window and click it. After that, the screenshot of the window will be immediately saved to the desktop in PNG format.

Resources about how to work with database in PHP:

PHP Data Objects (PDO): https://www.php.net/manual/en/book.pdo.php

Here are some specific topics that we used in this lab:

DB Connection: https://www.php.net/manual/en/pdo.connections.php

PDO — The PDO class

- PDO::setAttribute — Set an attribute
- PDO::beginTransaction — Initiates a transaction
- PDO::commit — Commits a transaction
- PDO::rollBack — Rolls back a transaction
- PDO::prepare — Prepares a statement for execution and returns a statement object

PDOStatement — The PDOStatement class

- PDOStatement::bindParam — Binds a parameter to the specified variable name
- PDOStatement::execute — Executes a prepared statement
- PDOStatement::fetch — Fetches the next row from a result set
- PDOStatement::fetchAll — Fetches the remaining rows from a result set
- PDOStatement::rowCount — Returns the number of rows affected by the last SQL statement

Work with Mysql in PHP

1. Let's get tables and data ready.

Please carefully read the SQL statement to understand the schema and data.

```
drop table if exists lab5_accounts;
drop table if exists lab5_customer;

create table lab5_customer(
username char(15) primary key,
password varchar(256) not null,
firstname char(20),
lastname char(20)
);

create table lab5_accounts(
username char(15),
account_no char(10),
balance decimal(10,2),
primary key (account_no),
foreign key ( username) references lab5_customer(username)
);
```

:

```
insert into lab5_customer values ('msmith', sha2("hi",256), 'Mary', 'Smith');
insert into lab5_customer values ('jdoe', sha2("nihao",256), 'John', 'Doe');
insert into lab5_accounts values ('msmith', 'A100', 500);
insert into lab5_accounts values ('msmith', 'B100', 5000);
insert into lab5_accounts values ('msmith', 'C100', 50000);
insert into lab5_accounts values ('jdoe', 'A101', 10000);
```

2. Get the login function to work with the real user and passwd in the database

2.1. Create a file **db.ini** to have the database connection information in it. Please don't put the db.ini under the classdb directory as people could access it via url. You may put it under your home dir. You do need to give read access to Apache like you did for other web files.

```
dsn = "mysql:host=classdb.it.mtu.edu;dbname=YOURDATABASENAME"
username = "youruser"
password = "yourpasswd"
```

2.2. Create a file db.php to include all the functions that are related to database.

Let's create two functions: connectDB() and authenticate().

- Please note authenticate () calls connectDB().
- Please pay attention how prepared statement is used.
- Please pay attention how to use fetch() to get the data from select statement

- Change the ..... in "/…….db.ini" to the correct directory name where you created db.ini. Use the absolute path which starts with /.

```php
<?php
function connectDB()
{
    $config = parse_ini_file("/…….db.ini");
    $dbh = new PDO($config['dsn'], $config['username'], $config['password']);
    $dbh->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    return $dbh;
}
//return number of rows matching the given user and passwd.
function authenticate($user, $passwd) {
    try {
        $dbh = connectDB();
        $statement = $dbh->prepare("SELECT count(*) FROM lab5_customer ".
                                   "where username = :username and password = sha2(:passwd,256) ");
        $statement->bindParam(":username", $user);
        $statement->bindParam(":passwd", $passwd);
        $result = $statement->execute();
        $row=$statement->fetch();
        $dbh=null;

        return $row[0];
    }catch (PDOException $e) {
        print "Error!" . $e->getMessage() . "<br/>";
        die();
    }
}
?>
```

2.3. Modify login.php to call authenticate ()

First we need to include the db.php in the login.php. At the beginning of the php code, add:

```
require "db.php";
```

Then change this line:

```
if ($_POST["username"]=="Mary" && $_POST["password"] =="Hello") {
```

to

```
if (authenticate($_POST["username"], $_POST["password"]) ==1) {
```

2.4. Use the username and password in the table lab5_customer to test your login.php to make sure it works as you expected.

3. Implement the function of displaying balance in bankoperation.php

In Lab4, after the user logged in successfully from login.php, they will be redirected to the main.php, where the user shall see the two buttons: Accounts and Transfer. We will implement these two functions in a new file **bankoperation.php**.

If "Accounts" button is clicked in main.php, display the account data for the user in bankoperation.php. Like this:

Account	Balance
A100	50.00
B100	150.00
C100	100.00

Follow the following steps to compete this function.

1) First, let's create a function get_accounts() in db.php to retrieve all data in database. Please pay attention how result set is fetched and how it is being used as return value.

```
function get_accounts($user)
{
    //connect to database
    //retrieve the data and display
    try {
        $dbh = connectDB();

        $statement = $dbh->prepare("SELECT account_no, balance name FROM lab5_accounts where username =
:username ");
        $statement->bindParam(":username", $user);
        $statement->execute();

        return $statement->fetchAll();
        $dbh = null;
    } catch (PDOException $e) {
        print "Error!" . $e->getMessage() . "<br/>";
        die();
    }
};
```

2) Set the form method to be post and action for "transfer" and "account" buttons in main.php to be *bankoperation.php*

3) Add code in bankoperation.php to display the data when "Accounts" is clicked.

```
<style>
table, th, td {
  border: 1px solid black;
  border-collapse: collapse;
}
</style>
```

```php
<?php
require "db.php";
if (isset($_POST["accounts"])) {
    $accounts = get_accounts($_SESSION["username"]);
?>

    <table>
    <tr>
    <th>Account</th>
    <th>Balanc</th>
    </tr>

    <?php
    foreach ($accounts as $row) {
        echo "<tr>";
        echo "<td>" . $row[0] . "</td>";
        echo "<td>" . $row[1] . "</td>";
        echo "</tr>";
    }
    echo "<table>";
}
?>
```

4) Modify bankoperation.php to join the session.

If no user is logged in yet, the user will be redirected to the login page.

```php
session_start();
if (!isset($_SESSION['username'])) {
    header("LOCATION:login.php");
}
```

4. Implement the transfer function

4.1 Add code in bankoperations.php to display the form when “Transfer” button is clicked.

4.2 Add code in bankoperations.php to process the transfer function when “Confirm” is clicked.

```php
if (isset($_POST["confirm"])) {
    $from = $_POST["from_account"];
    $to = $_POST["to_account"];
    $amount = $_POST["amount"];
    $user = $_SESSION["username"];
    transfer($from, $to, $amount, $user);
}
```

4.3 Implement function transfer()in db.php.

Please pay attention how transaction is used. Feel free to add more checking inside the transaction.

```php
function transfer($from, $to, $amount, $user)
{
    try {
        $dbh = connectDB();
        $dbh->beginTransaction();

        // check if there are enough balance in the from account
        $statement = $dbh->prepare("select balance from lab4_accounts where account_no=:from ");
        $statement->bindParam(":from", $from);
        $result = $statement->execute();
        $row = $statement->fetch();
        if ($row) {
            $currentBalance = $row[0];
            if ($currentBalance < $amount) {
                $dbh->rollBack();
                $dbh = null;
                return "Not enough balance in $from";;
            }
        } else {
            $dbh->rollBack();
            $dbh = null;
            return "Account $from does not exist";
        }

        $statement = $dbh->prepare("update lab4_accounts set balance = balance - :amount " .
            "where account_no=:from");
```

```
        $statement->bindParam(":amount", $amount);
        $statement->bindParam(":from", $from);
        $result = $statement->execute();
        $rowCount = $statement->rowCount();
        if ($rowCount != 1) {
            $dbh->rollback();
            return  "Something is not right because the total number of rows that will be affected is " .
$rowCount;
        }

        $statement = $dbh->prepare("update lab4_accounts set balance = balance + :amount " .
            "where account_no= :to");
        $statement->bindParam(":amount", $amount);
        $statement->bindParam(":to", $to);
        $result = $statement->execute();
        $rowCount = $statement->rowCount();
        if ($rowCount != 1) {
            $dbh->rollback();
            return  "Something is not right because the total number of rows that will be affected is " .
$rowCount;
        }

        $dbh->commit();
        return "Money has been transfered successfully";
    } catch (Exception $e) {
        $dbh->rollBack();
        echo "Failed: " . $e->getMessage();
    }
}
```

4.4 Test and make sure everything works as expected.